รีวิว

สุทธิศักดิ์ พงศ์ธนาพาณิช
(sutthisakp@yahoo.com)

แนะนำ Microsoft Visual Basic 6.0

แนะนำ Visual Basic 6.0 และคุณสมบัติใหม่ ๆ เพื่อเป็นแนวทางในการจะอัพเกรดซอฟต์แวร์

เมื่อครั้งที่ไมโครซอฟท์ได้มีการเปิดตัววินโดวส์ 95 ชุดเครื่องมือในการพัฒนาระบบงาน Visual Studio 97 ก็ได้ถูกนำเสนอต่อผู้ใช้ในรูปแบบของชุดรวมเครื่องมือในการพัฒนาระบบงาน ซึ่งเป็นกลยุทธ์ที่ไมโครซอฟท์เคยประสพความสำเร็จมาแล้วจากการนำชุดซอฟต์แวร์ทั่วไปมารวมกันในชื่อไมโครซอฟท์ออฟฟิศ 97 และเครื่องมือตัวหนึ่งที่ได้ถูกนำมารวมใน Visual Studio 97 ก็คือ Visual Basic 5.0 ซึ่งเป็นเวอร์ชันที่ได้รับความชื่นชมจากบรรดาโปรแกรมเมอร์มากที่สุด เพราะเป็นเครื่องมือที่มีความลงตัวระหว่างความเร็วของโปรแกรม .exe และขนาดของเอนไวรอนเมนต์ที่ใช้ทรัพยากรของระบบไม่มากนัก และยังทำงานได้ดีกับฮาร์ดแวร์ที่ใช้เพนเทียมรุ่นเก่า ๆ

ในราว ๆ กลางปี 2541 นี้เอง ไมโครซอฟท์ได้ทำการปรับปรุงระบบปฏิบัติการวินโดวส์สำหรับใช้งานในระดับครอบครัวหรือส่วนบุคคล ภายใต้ชื่อวินโดวส์ 98 เพื่อลดแรงเสียดทานจากผู้ใช้ที่เกิดจากความไม่มีเสถียรภาพของวินโดวส์ 95 และในขณะเดียวกันได้มีการเปิดตัว Visual Studio 6.0 เพื่อสนับสนุนการพัฒนาซอฟต์แวร์ที่สามารถใช้คุณสมบัติใหม่ ๆ ของวินโดวส์ 98 ได้อย่างสมบูรณ์ สำหรับ Visual Basic ก็ได้รับการพัฒนามาเป็น Visual Basic 6.0 โดยจากข้อมูลเบื้องต้นที่ผู้เขียนได้ทำการศึกษา พบว่าไมโครซอฟท์ได้ทำการแก้ไขข้อผิดพลาดที่เกิดขึ้นกับเอนไวรอนเมนต์และองค์ประกอบของ Visual Basic 5.0 กว่า 150 รายการ โดยก่อนหน้าที่ไมโครซอฟท์จะเปิดตัว Visual Basic 6.0 ก็ได้ทำการแก้ไขข้อผิดพลาดไปบ้างแล้วบางประการในรูปของ Service Pack (สามารถดาวน์โหลดได้จากเว็บไซท์ของไมโครซอฟท์ http://www.microsoft.com)

สำหรับในบทความตอนนี้ผู้เขียนก็จะทำการแนะนำ Visual Basic 6.0 และคุณสมบัติใหม่ ๆ รวมทั้งที่ได้รับการปรับปรุงจากเวอร์ชันที่ผ่านมา เพื่อเป็นแนวทางสำหรับผู้อ่านที่จะตัดสินใจในการอัพเกรดซอฟต์แวร์

มีอะไรใหม่ใน Visual Basic 6.0

สำหรับตัวผู้เขียนเมื่อได้ทดสอบ Visual Basic 6.0 (โดยเปรียบเทียบกับ Visual Basic 5.0) พบว่าถ้าหากเราไม่ได้ใช้ Visual Basic เพื่อพัฒนาแอพพลิเคชันที่เกี่ยวข้องกับงานด้านฐานข้อมูลหรืออินเทอร์เน็ต เช่น การสร้าง HTLM

แบบไดนามิกที่สามารถใช้ได้บน Internet Explotrer 4.0 แล้วละก็ รู้สึกว่า Visual Basic 6.0 ให้ความแตกต่างจาก Visual Basic 5.0 น้อยกว่าที่ Visual Basic 5.0 ให้ความแตกต่างจาก Visual Basic 4.0 โดยเฉพาะในความเร็วของไฟล์ .exe ที่ได้รับการคอมไพล์ในรูปของโค้ดแท้ (native code) กลับพบว่า Visual Basic 6.0 ได้รับการปรับปรุงในส่วนนี้ให้ดีขึ้นกว่า Visual Basic 5.0 น้อยมาก และแทบจะไม่ให้ความรู้สึกที่แตกต่างเมื่อนำไฟล์ .exe ไปทดสอบบนเครื่องระดับ Pentium II แต่อย่างไรก็ตาม Visual Basic 6.0 ก็มีส่วนที่ได้รับการปรับปรุงเพิ่มเติมพอสมควรทีเดียว ซึ่งสามารถแยกออกเป็นคุณสมบัติหลัก ๆ ดังนี้

การเข้าถึงข้อมูลด้วยเทคโนโลยี OLE

Visual Basic เป็นเครื่องมือในการพัฒนาซอฟต์แวร์ที่ได้รับการพูดถึงในเรื่องของจุดด้อยในด้านการเข้าถึงฐานข้อมูลเป็นอย่างมาก ซึ่งไมโครซอฟท์ได้เริ่มที่จะปรับปรุงคุณสมบัติด้านการเข้าถึงฐานข้อมูลให้กับ Visual Basic อย่างจริงจังเมื่อเวอร์ชัน 5.0 ที่ผ่านมานี้เอง และสำหรับเวอร์ชัน 6.0 คุณสมบัติดังกล่าวก็ได้รับการปรับปรุงเป็นอย่างมากทีเดียว และถือได้ว่ากลายมาเป็นจุดเด่นที่สำคัญของ Visual Basic ไปทันที เพราะนอกจากตัวของ Visual Basic ยังจะมีเครื่องมือที่ช่วยในการเชื่อมโยงฐานข้อมูลไม่ว่าจะเป็น Microsoft SQL Server, Oracle 7.3.3 หรือ AS/400 เข้ากับแอพพลิเคชันได้อย่างสมบูรณ์ ซึ่งเทคโนโลยีที่ Visual Basic สนับสนุนมีดังนี้

ADO (ActiveX Data Objects)

เทคโนโลยีการเข้าถึงข้อมูล ที่ช่วยให้การอินเทอร์เฟสระดับแอพพลิเคชันทำได้อย่างง่ายและสะดวก โดยเฉพาะข้อมูลภายใต้สถาปัตยกรรม OLE DB ไม่ว่าจะเป็นฐานข้อมูลทั้งแบบ relational และไม่สนับสนุนคุณสมบัติ relational ระบบไฟล์อีเมล์ หรือกราฟิกต่าง ๆ ซึ่งได้รับการปรับปรุงให้เข้ากันได้ดีกับ Visual Basic 6.0

Data Environment

ตัวออกแบบสภาพแวดล้อมข้อมูล (Data Environment) สำหรับ Visual Basic 6.0 เป็นอีกคุณสมบัติที่ผู้เขียนติดใจมากทีเดียว เมื่อเทียบกับเวอร์ชันที่ผ่านมาที่ใช้ตัวออกแบบ ActiveX UserConnection สำหรับสร้างการเข้าถึงฐานข้อมูลแบบ Remote Data Objects (RDO) เนื่องจาก Data Environment ช่วยให้ผู้อ่านสามารถสร้างการเข้าถึงฐานข้อมูลในขณะรันแอพพลิเคชัน (run-time) ที่สามารถโปรแกรมได้ และสามารถใช้เชื่อมกับคอนโทรล (หมายถึงใช้เป็นแหล่งข้อมูลโดยตรง) ได้อีกด้วย

ADO Data Control

คอนโทรลข้อมูล ADO ที่สนับสนุนข้อมูล OLE DB ซึ่งช่วยให้คอนโทรลด้านฐานข้อมูลของ Visual Basic สามารถติดต่อกับตัวให้บริการฐานข้อมูลภายใต้มาตรฐาน OLE DB ของไมโครซอฟท์ ซึ่งจะช่วยลดการเขียนโค้ดลงไปได้มากทีเดียว นอกจากนี้คอนโทรล DataList และ DataCombo ยังได้รับการปรับปรุงให้สนับสนุน OLE DB อีกด้วย ซึ่งสามารถใช้งานร่วมกับคอนโทรลข้อมูล ADO ได้อย่างไม่มีปัญหา

สนับสนุน OLE DB

OLE DB เป็นอินเทอร์เฟสระดับล่างที่ช่วยให้การเข้าถึงฐานข้อมูลในหลาย ๆ รูปแบบสามารถกระทำได้โดยไม่จำเป็นต้องเขียนโค้ดแยกตามรูปแบบต่าง ๆ แต่ OLE DB จะช่วยในการแปลงรูปแบบให้อยู่ในมาตรฐานเดียวกัน สำหรับระดับแอพพลิเคชันจะสามารถนำไปใช้ได้

Hierarchical FlexGrid control

คอนโทรล Hierarchical FlexGrid (MSHFlexGrid) ช่วยให้ผู้อ่านสามารถแสดงผลฐานข้อมูลได้ด้วยรูปแบบและลูกเล่นที่มากกว่าคอนโทรล FlexGrid ของเวอร์ชันที่ผ่านมา และคอนโทรล Hierarchical FlexGrid เป็นคอนโทรลประเภทไม่สามารถแก้ไขข้อมูล ดังนั้นข้อมูลต้นฉบับที่นำมาแสดงผลจึงปลอดภัยจากการแก้ไขอย่างสมบูรณ์

งานด้านอินเทอร์เน็ต

Visual Basic ได้เริ่มมีบทบาทกับการเขียนแอพพลิเคชันสำหรับงานด้านอินเทอร์เน็ตครั้งแรก เมื่อไมโครซอฟท์ได้พัฒนา VBScript เพื่อแข่งขันกับ JavaScript แต่สำหรับ Visual Basic 6.0 ก็ได้รับการปรับปรุงความสามารถสำหรับงานด้านอินเทอร์เน็ตไปมากทีเดียว โดยที่มีความสำคัญ ๆ มีดังนี้

แอพพลิเคชัน IIS

แอพพลิเคชัน IIS เป็นแอพพลิเคชันด้านอินเทอร์เน็ตที่ทำงานบนเว็บเซอร์ฟเวอร์ ซึ่งจะใช้หลักการสร้างไฟล์ HTML แบบไดนามิกโดยการใช้ Visual Basic 6.0 สร้างโมดูลที่ถูกคอมไพล์สำหรับใช้ประมวลผลต่าง ๆ โดยให้มีความเชื่อมโยงกับ HTML ซึ่งจะช่วยให้ HTML ที่ได้มีความเป็นปฏิสัมพันธ์กับผู้ใช้มากขึ้น

แอพพลิเคชัน DHTML

HTML แบบไดนามิกที่สามารถทำงานได้บน Internet Explorer 4.0 ซึ่งสนับสนุนเว็บเพจที่มีทั้งคุณสมบัติ เหตุการณ์ และวิธี เช่นเดียวกับความสามารถของคอนโทรล ActiveX ทั่ว ๆ ไปของ Visual Basic ซึ่งผู้อ่านจะสามารถสร้าง HTML แบบไดนามิกที่สามารถทำได้โดยไม่ต้องอาศัยเว็บเซอร์ฟเวอร์

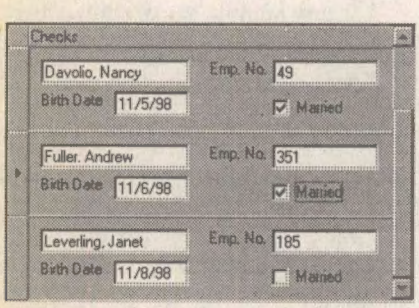
AsyncRead Enhancements

ขยายขีดความสามารถในกระบวนการปฏิบัติงานแบบไม่สอดคล้อง (asynchronous) ที่สามารถรายงานสถานะปัจจุบันของการปฏิบัติงานต่าง ๆ ได้มากมาย เช่น ความคืบหน้าและสถานะของการปฏิบัติงาน จำนวนไบต์ที่อ่านได้ล่าสุด เป็นต้น

เพิ่มเติมคอนโทรล ActiveX ใหม่

คอนโทรล CoolBar คอนโทรล CoolBar เป็นคอนโทรลตัวบรรจุที่ใช้ในการสร้างแถบเครื่องมือแบบ user-configurable เช่นเดียวกับที่ปรากฏใน Internet Explorer ของไมโครซอฟท์

คอนโทรล DataRepeater เป็นคอนโทรลที่ใช้สร้างคอนโทรลสำหรับแสดงผลฐานข้อมูลในรูปแบบที่ผู้อ่านต้องการ เช่น การแสดงผลทะเบียนประวัติทั้งหมดของพนักงานแต่ละคน ดังในรูปที่ 1 ซึ่งผู้อ่านสามารถสร้างได้โดยการแทรกองค์ประกอบ UserControl ลงใน DataRepeater เท่านั้น จากนั้นก็แทรกคอนโทรลที่ต้องการใช้แสดงผลลงใน User Control เพื่อสร้างฟิลด์สำหรับการแสดงผลตามที่ต้องการ



รูปที่ 1 แสดงวิธีการสร้างรูปแบบการแสดงผลฐานข้อมูลด้วย DataRepeater

คอนโทรล DatetimePicker แสดงผลวันที่และเวลา ทั้งในรูปแบบของข้อความและปฏิทินรายเดือน (drop-down) ซึ่งช่วยให้ผู้อ่านสามารถเลือกหรือแก้ไขปฏิทินได้ง่าย

รูปที่ 2 แสดงคอนโทรล DateTimePicker

คอนโทรล MonthView แสดงผลวันที่ในแต่ละเดือน คล้าย ๆ กับปฏิทินที่เราใช้กันอยู่ทั่ว ๆ ไป ซึ่งช่วยให้ผู้อ่านสามารถวันที่หรือเดือนได้ง่ายมาก โดยการคลิกในพื้นที่ของคอนโทรล

นอกจากนี้ยังมีคอนโทรลอื่น ๆ อีกมากมายที่ได้รับการปรับปรุง ไม่ว่าจะเป็น

รูปที่ 3 แสดงคอนโทรล MonthView

คอนโทรลภายใน (intrinsic) หรือคอนโทรลร่วมวินโดวส์ โดยสำหรับ Visual Basic 6.0 สามารถเชื่อมต่อกับคอนโทรลร่วมวินโดวส์ได้ทั้ง 3 ชุด (Windows Common Controls 6.0, -2 6.0 และ -3 6.0)

การปรับปรุงด้านภาษา

นอกจากการปรับปรุงในส่วนขององค์ประกอบต่าง ๆ แล้ว Visual Basic 6.0 ยังได้รับการปรับปรุงในส่วนของภาษาอีกด้วย ซึ่งในหลายส่วนได้รับการเพิ่มความสามารถจนอยู่ในระดับที่น่าพอใจพอสมควรทีเดียว

ฟังก์ชันสามารถคืนกลับข้อมูลอะเรย์ได้

แต่เดิมถ้าหากผู้อ่านต้องการส่งคืนข้อมูลในรูปแบบอะเรย์กลับจากโพรซีเดอร์ ผู้อ่านต้องเลือกที่จะใช้โพรซีเดอร์แบบ Sub เท่านั้น และใช้วิธีการส่งกลับข้อมูลอะเรย์ในรูปแบบของการส่งพารามิเตอร์เท่านั้น ดังโค้ดตัวอย่างต่อไปนี้ สมมติว่าต้องการสร้างโพรซีเดอร์ที่รายงานผลการบวก ลบ คูณ และหารตัวเลข 2 ตัว

```
Sub SendArrbyParm(X1 As Double, _
    X2 As Double, Arr() As Double)
    Arr(1) = X1 + X2
    Arr(2) = X1 - X2
    Arr(3) = X1 * X2
    Arr(4) = X1 / X2
End Sub
```

แต่สำหรับ Visual Basic 6.0 ผู้อ่านไม่จำเป็นต้องส่งอะเรย์ผ่านทางพารามิเตอร์เพื่อรับผลลัพธ์จากโพรซีเดอร์อีกต่อไป แต่ผู้อ่านสามารถใช้ฟังก์ชันที่รายงานค่าแบบอะเรย์ได้โดยตรงดังนี้

```
Function SendArrbyFunc(X1 As Double,_
        X2 As Double) As Double()
    Dim TmpArr(1 To 4) As Double

    TmpArr(1) = X1 + X2
    TmpArr(2) = X1 - X2
    TmpArr(3) = X1 * X2
    TmpArr(4) = X1 / X2
    SendArrbyFunc = TmpArr
End Function
```

สามารถกำหนดอะเรย์ให้กับอะเรย์ได้โดยตรง

ในการคัดลอกทุก ๆ สมาชิกของอะเรย์หนึ่งให้กับอีกอะเรย์หนึ่งในเวอร์ชันที่ผ่าน ๆ มา ผู้อ่านต้องทำการคัดลอกทีละสมาชิกโดยกระทำภายในวงลูป เช่น For...Next เป็นต้น

```
ReDim Arr0(0 To 10)
ReDim Arr1(Lbound(Arr0) To _
      UBound(Arr0)
    For i - Lbound(Arr0) To _
Ubound(Arr0)
    Arr1(i) = Arr0(i)
Next
```

แต่สำหรับ Visual Basic 6.0 ผู้อ่านกำหนดชื่อของอะเรย์ให้กันได้โดยตรง ซึ่ง Visual Basic 6.0 จะทำการคัดลอกทุก ๆ สมาชิกให้ทันทีโดยอัตโนมัติ

```
ReDim Arr0(0 To 10)
ReDim Arr1(Lbound(Arr0) To _
      UBound(Arr0)

Arr1 = Arr0
```

ซึ่งโดยอาศัยวิธีใหม่จะช่วยให้แอพพลิเคชันที่ได้ ทำงานรวดเร็วขึ้นอีกด้วย

CallByName

Visual Basic 6.0 อนุญาตให้ผู้อ่านสามารถเขียนโค้ดเพื่อเรียกคุณสมบัติหรือวิธีของออบเจ็กต์หรือคอนโทรลได้โดยตรงโดยอาศัยฟังก์ชัน CallByName แทนที่จะใช้วิธีการเรียกโดยการกำหนดคุณสมบัติ หรือวิธีของคอนโทรล ตามหลังชื่อของคอนโทรล ดังเช่นโค้ด 2 บรรทัดต่อไปนี้จะให้ผลในทางปฏิบัติที่เหมือนกันทุกประการ สำหรับรูปแบบของการเรียกใช้ฟังก์ชัน CallByName มีดังนี้

```
CallByName(object, procedurename,
calltype,[arguments()])

CallByName Text1, "Move", vbMethod, 100,
100
      หรือ
Text1.Move 100, 100
```

ฟังก์ชัน StrConv ที่สนับสนุนหลายภาษา

ฟังก์ชัน StrConv ของ Visual Basic 6.0 จะมีพารามิเตอร์ LCID ที่สนับสนุนระบบปฏิบัติการแบบหลายภาษา โดยที่ผู้อ่านสามารถกำหนดค่า LocaleID สำหรับสตริงที่แตกต่างกับค่า LocaleID ของระบบได้ ซึ่งมีรูปแบบการเรียกใช้ดังนี้

StrConv(string, conversion, LCID)

นอกจากนี้ยังมีฟังก์ชันอื่น ๆ ที่ได้รับการเพิ่มเติมอีกมากมาย เช่น ฟังก์ชัน StrReverse ที่ช่วยกลับลำดับของตัวอักษรในสตริง ฟังก์ชัน Join ที่ช่วยในการเชื่อมสตริงเข้าด้วยกัน เป็นต้น

การอัพเกรดคอนโทรลร่วมวินโดวส์

คอนโทรลร่วมวินโดวส์เป็นคอนโทรลที่แอพพลิเคชันต่าง ๆ ของวินโดวส์มักจะใช้งานร่วมกัน สำหรับผู้อ่านที่ต้องการใช้ Visual Basic 6.0 ขอแนะนำให้ทำการอัพเกรดคอนโทรลร่วมวินโดวส์เวอร์ชัน 6.0 แทนเวอร์ชันที่มากับ Visual Basic 5.0 เพราะนอกจากจะง่ายต่อการสร้างชุดติดตั้งแล้ว ยังสามารถใช้คอนโทรลใหม่ ๆ ได้อีกด้วย สำหรับไฟล์ที่จำเป็นของคอนโทรลร่วมวินโดวส์ทั้งสองเวอร์ชันมีดังนี้

Visual Basic 5.0	Visual Basic 6.0
ComCtl32.ocx	MsComctl.ocx
ComCt232.ocx	MsComct2.ocx
ComCtl32.dll	

อย่างไรก็ตาม คอนโทรลร่วมวินโดวส์เวอร์ชัน 6.0 จะมีขนาดรวมที่ใหญ่กว่าเวอร์ชัน 6.0 มากทีเดียว โดยในตารางต่อไปนี้ผู้เขียนได้เปรียบเทียบขนาดของไฟล์ที่เกี่ยวข้องทั้งหมด

Windows Common control	File Size (KB)
Visual Basic 5.0	*1,128*
ComCtl32.ocx	596
ComCt232.ocx	161
ComCtl32.dll	371
Visual Basic 6.0	*1,668*
MsComctl.ocx	1038
MsComct2.ocx	630

สำหรับคอนโทรลร่วมวินโดวส์ทั้งหมดที่มากับ Visual Basic 6.0 ซึ่งได้ถูกจัดเก็บไว้ในไฟล์ .ocx ทั้งสองตามที่ปรากฏในตารางข้างต้นมีดังนี้

File name	Common Control
MsComctl.ocx	ImageCombo ImageList ListView ProgressBar Slider StatusBar Toolbar TabStrip TreeView
MsComct2.ocx	Animation DateTimePicker FlatScrollBar MonthView UpDown

ข้อผิดพลาดที่ได้รับการแก้ไขใน Visual Basic 6.0

นอกจากคุณสมบัติใหม่และคอนโทรลส่วนที่ได้รับการปรับปรุงให้สอดคล้องกับเทคโนโลยีใหม่ของไมโครซอฟท์แล้ว Visual Basic 6.0 ยังได้รับการแก้ไขข้อผิดพลาด (bug) จากเวอร์ชันที่ผ่านมา (โดยเฉพาะจากเวอร์ชัน 5.0) อีกมากมาย โดยในที่นี้ผู้เขียนขอสรุปเฉพาะส่วนที่สำคัญและมักพบบ่อยในการเขียนโปรแกรม ดังนี้

ปัญหาปี 2000 (Y2K)

Visual Basic 6.0 ได้รับการแก้ไขปัญหาเรียบร้อยแล้วครับ ดังนั้นผู้อ่านที่ใช้ Visual Basic 6.0 ในการพัฒนาแอพพลิเคชันที่ต้องทำงานในด้านของเซอร์ฟเวอร์ ไม่ว่าจะเป็นเซอร์ฟเวอร์ฐานข้อมูลหรือเว็บเซอร์ฟเวอร์ ก็ไม่ต้องกังวลกับปัญหา Y2K อีกต่อไป (ข้อมูลเพิ่มเติมสามารถตรวจสอบได้จากเว็บไซท์ของไมโครซอฟท์

```
http://www.microsoft.com/y2k/)
```

ข้อผิดพลาดในคอนโทรล ListView

คอนโทรล ListView ที่มากับ Visual Basic 5.0 จะเกิดข้อผิดพลาด (run-time 35600) ถ้าหากแอพพลิเคชันดังกล่าวทำงานบนระบบที่มีไฟล์ COMCTL32.OCX เวอร์ชัน 5.00.3714 และนอกจากนี้ถ้าหากแอพพลิเคชันดังกล่าวถูกคอมไพล์โดยอ้างอิงถึงไฟล์ COMCTL32.OCX เวอร์ชัน 5.00.3422 หรือ 5.00.3601 ก็จะชน (crash) กับระบบทันที ถ้าหากทำงานบนระบบที่มีไฟล์ COMCTL32.OCX เวอร์ชัน 5.00.3714 ซึ่งเป็นปัญหาความไม่คอมแพทิเบิลระดับไบนารีของโค้ดนั่นเอง

ข้อผิดพลาดในคอนโทรล DBGrid

แอพพลิเคชันจะชน (crash) กับระบบทันที ถ้าหากมีการปรับขนาดของคอลัมน์ (resize) ที่ถูกแยกส่วนด้วยวิธี Split ของคอนโทรล DBGrid ซึ่งข้อผิดพลาดนี้เป็นข้อผิดพลาดในระดับโค้ดไบนารีของคอนโทรล DBGrid เลยทีเดียว (ผู้เขียนเข้าใจว่าอย่างนั้นครับ)

เกิด GPF เมื่อกำหนดฟอนต์ให้คอนโทรล

ปัญหาที่รุนแรงนี้จะเกิดขึ้นกับแอพพลิเคชันแบบหลายเอกสาร (MDI) เมื่อหน้าต่างคุณสมบัติอยู่ในสถานะเป็นหน้าต่าง (ไม่เรียงต่อเนื่อง) และผู้อ่านเลือกหลาย ๆ คอนโทรล และพยายามแก้ไขฟอนต์ให้กับคอนโทรลเหล่านั้นโดยการแก้ไขจากคุณสมบัติ Font ของหน้าต่างคุณสมบัติ

คอนโทรล MCI32.OCX

คอนโทรล MCI32.OCX ที่มากับ Visual Basic 5.0 จะไม่ทำการสร้างไฟล์เพื่อบันทึกเสียงในรูปแบบไฟล์ .wav เมื่อเรียกใช้โดยคำสั่ง Command

คอนโทรล UpDown

คุณสมบัติ Enabled ไม่สามารถกำหนดสถานะ disabled เมื่อคุณสมบัติ Enabled มีค่าเท่ากับ False

เอนไวรอนเมนต์สำหรับแอพพลิเคชันแบบ SDI

หน้าต่างแก้ไขโค้ดโมดูล (.bas) จะปรากฏเป็นหน้าต่างแบบโปร่งใส ซึ่งจะมองเห็นเฉพาะกรอบของหน้าต่างเท่านั้น เมื่อมีการสร้างแอพพลิเคชันแบบเอกสารเดี่ยว (SDI)

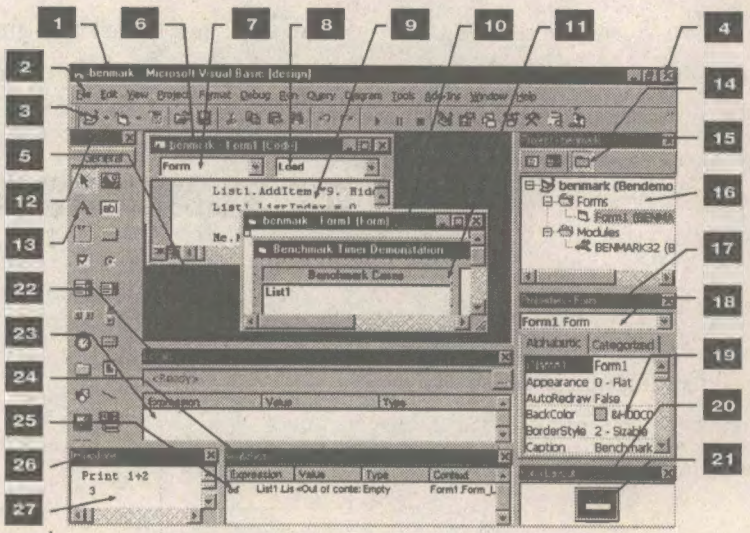
คอนโทรล Toolbox

แอพพลิเคชันจะชนกับระบบ (ผู้เขียนพบเพียงบางครั้งเท่านั้น) และเมื่อผู้อ่านเลื่อนเคอร์เซอร์มาอยู่บนคอนโทรล Toolbox ในขณะรันแอพพลิเคชัน เนื่องจากเกิดการเข้าถึงบางส่วนของระบบแบบไม่ปลอดภัย (Access Violation ซึ่งรายงานโดยโปรแกรมแก้ไขการชนของระบบชื่อ Norton CrashGuard 3.0)

โปรแกรม Visual SourceSafe

โปรแกรม Visual SourceSafe ทำให้เกิดข้อผิดพลาดเมื่อทำการอ่านโปรเจ็กต์กลับคืนโดย Visual Basic 5.0 ซึ่งผู้เขียนคาดว่าน่าจะเป็นข้อผิดพลาดที่โปรแกรม Visual SourceSafe มากกว่าจะเกิดจาก Visual Basic 5.0 ซึ่งข้อผิดพลาดที่พบบ่อยก็คือ Application error

แต่เป็นที่ทราบกันดีว่า ไม่ว่าจะเป็น Visual Basic เวอร์ชันใด เมื่อมีผู้ใช้ไประยะหนึ่งก็จะพบข้อผิดพลาดตามมาเรื่อย ๆ ดังนั้นสำหรับ Visual Basic 6.0 ก็เช่นกัน เท่าที่ทราบในปัจจุบันก็มีผู้พบข้อผิดพลาดบ้างแล้ว และคาดว่าทางไมโครซอฟท์คงจะทำการแก้ไขและอัพเดตให้ในรูปแบบของ Service Pack ตามเคย ในอนาคตอันใกล้นี้ครับ ดังนั้นถ้าหากผู้อ่านไม่รีบร้อนที่จะลองของใหม่ ผู้เขียนก็ขอแนะนำให้คอยสักระยะจนกว่า Visual Basic จะได้รับการแก้ไขอีกสักหน่อย น่าจะปลอดภัยและน่าใช้งานกว่าครับ



รูปที่ 4 แสดงเอนไวรอนเมนต์ของ Visual Basic 6.0

รายละเอียดของแต่ละหมายเลข

1) แถบชื่อแอพพลิเคชัน (title bar)
2) แถบเมนู (menu bar)
3) แถบเครื่องมือ (toolbar)
4) ปุ่มเปลี่ยนขนาดหน้าต่าง ซึ่งประกอบด้วยปุ่ม Minimize ปุ่ม Maximize และปุ่ม Restore
5) พื้นที่ของเอนไวรอนเมนต์ Visual Basic
6) หน้าต่างแก้ไขโค้ด (code window)
7) ช่องเลือกรายการชื่อฟอร์ม คอนโทรล หรือออบเจ็กต์ที่บรรจุในโปรเจ็กต์ที่กำลังแก้ไข
8) ช่องเลือกรายการชื่อเหตุการณ์ (event procedure) ของรายการที่ถูกเลือกจากลำดับหมายเลข 7 ข้างต้น
9) พื้นที่แก้ไขโค้ด ซึ่งผู้อ่านสามารถแบ่งพื้นที่แก้ไขโค้ดออกเป็น 2 ส่วน (ส่วนบนและส่วนล่าง) โดยการใช้เมาส์ลาก splitter ซึ่งเป็นปุ่มสี่เหลี่ยมเล็ก ๆ บนปุ่มลูกศรเลื่อนขึ้นในแถบเลื่อน (scroll bar) ที่ด้านขวาของหน้าต่างแก้ไขโค้ด
10) หน้าต่างแสดงผลฟอร์ม (form window) ในขณะออกแบบ สำหรับการแก้ไขหรือปรับแต่ง
11) พื้นที่แสดงผลฟอร์ม
12) หน้าต่างกล่องเครื่องมือ (toolbox window)
13) คอนโทรล (control) ทั้งหมดที่ถูกอ้างอิงให้กับโปรเจ็กต์ที่กำลังแก้ไขในขณะนั้น
14) หน้าต่างโปรเจ็กต์ (project explorer)
15) แถบเลือกรูปแบบการแสดงผลเป็นฟอร์ม โค้ดหรือสลับวิธีแสดงรายการในหน้าต่างโปรเจ็กต์
16) ช่องเลือกรายการฟอร์ม โมดูล คลาสโมดูล หรืออื่น ๆ
17) หน้าต่างคุณสมบัติ (property window)
18) ช่องเลือกรายการฟอร์มหรือคอนโทรล
19) ช่องเลือกรายการคุณสมบัติและค่าที่เกี่ยวข้อง
20) หน้าต่างจัดตำแหน่งแสดงผลฟอร์ม (form layout window)
21) รูปสำหรับลากไป-มา เพื่อจัดวางตำแหน่งฟอร์มบนจอภาพ
22) หน้าต่างแสดงผลสถานะตัวแปรภายในโพรซีเดอร์ (locals window)
23) พื้นที่แสดงรายการสถานะตัวแปรภายในโพรซีเดอร์ตรงตำแหน่งที่โปรแกรมหยุดการทำงานหรือหยุดการทำงานชั่วคราว (break)
24) หน้าต่างแสดงรายการสถานะตัวแปรที่ถูกกำหนดสถานะให้เป็นตัวแปรที่ถูกจับตามอง (watch window)
25) พื้นที่แสดงรายการสถานะตัวแปรที่ถูกกำหนดสถานะให้เป็นตัวแปรที่ถูกจับตามอง เมื่อถูกประมวลผล (watch variable)
26) หน้าต่างปฏิบัติโค้ดทันที (immediate window)
27) พื้นที่ปฏิบัติโค้ดทันที ซึ่งผู้อ่านสามารถพิมพ์โค้ดที่ต้องการประมวลผล และให้กด Enter เพื่อจบบรรทัด จากนั้น Visual Basic จะประมวลผลโค้ดบรรทัดดังกล่าวทันที